Google Web Images Groups News Froogle Local more »
compiler annotations memory locations Search Advanced Search Preferences

Web Results **1** - **10** of about **205,000** for **compiler annotations memory locations**. (**0.34** seconds)

Compiler Implementation Notes
The use of these **memory locations** managed in this way can be as fast as using ... The **Annotations** module written in Standard ML is used extensively for ...
www.smlnj.org/**compiler**-notes/ - 8k - Cached - Similar pages

LCLint User's Guide - Section 5: **Memory** Management
Its **location** in **memory** is used, but not its value. ... With implicit **annotations** (on by default), turning on **memory** checking for an unannotated program will ...
www.doc.ic.ac.uk/lab/cplus/lclint/sec5.html - 19k - Cached - Similar pages

[PPT] Static Analysis for Architecture Implementation Conformance in ...
File Format: Microsoft Powerpoint 97 - View as HTML
Never accesses a **memory location** outside its data area ... Ada-SPARK - **Compiler annotations** to make explicit data-flow information in a restricted subset of ...
www-rtsl.cs.uiuc.edu/~kowshik/files/Prelim.ppt - Similar pages

Inside LabVIEW - How the **Compiler** Works - Articles - NI Developer ...
move the value in register edx to the **memory location** ebp-8. Figure 5. Assembly code with **Annotations**. If you are hungry for more, the following link shows ...
www.ni.com/devzone/lvzone/dr_vi_archived6.htm - 28k - Sep 13, 2005 - Cached - Similar pages

[PDF] Classification of Code **Annotations** and Discussion of **Compiler** ...
File Format: PDF/Adobe Acrobat - View as HTML
location. The other categories of code **annotations** are. described in the following subsections. ... For example, the **compiler** may know the **memory** ...
artist.cs.uni-sb.de/WCET05/Papers/Kirner.pdf - Similar pages

[PS] Annotating the Java Bytecodes in Support of Optimization Joseph ...
File Format: Adobe PostScript - View as Text
However, a[i] and b[i] may refer to the same **memory location**. From these **annotations**, an annotation-aware JIT **compiler** can produce the same optimal native ...
campus.lakeforest.edu/~hummel/papers/java-cpe-98.ps - Similar pages

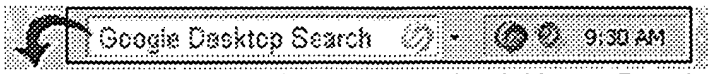
[PS] A Certifying **Compiler** for Java Christopher Colby Peter Lee George ...
File Format: Adobe PostScript - View as Text
Except for the **annotations** the output of the **compiler** is similar to that of conventional ... This does not include the **memory locations** that are known to be ...
raw.cs.berkeley.edu/Papers/touchstone_pldi00.ps - Similar pages

United States Patent Application: 0040010783
[0029] Once the **compiler** identifies the **memory** footprints, the **compiler** ... The **annotations** correspond to **memory** footprints of various blocks of the code ...
appft1.uspto.gov/.../ 20040010783&RS=DN/20040010783 - 41k - Cached - Similar pages

[PS] Chapter 1 An Overview of the SUIF **Compiler** for Scalable Parallel ...
File Format: Adobe PostScript - View as Text
... passes by **annotations** on the **compiler's** internal representation of the code. ... Interference occurs whenever different **memory locations** contend for the ...
cag-www.lcs.mit.edu/~saman/papers/siam95c.ps - Similar pages

C++ **Annotations** Version 6.2.2
In contrast, the C++ **compiler** knows exactly which in- or output action to perform given which ... At some **memory location** an int value is increased by one. ...
www.icce.rug.nl/documents/cplusplus/cplusplus03.html - 65k - Sep 12, 2005 - Cached - Similar pages

Gooooooooooogle ▶
Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**



Free! Instantly find your email, files, media and web history. Download now.

compiler annotations memory locatic Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google

©2005 Google

PORTAL
USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
compiler annotation "specific memory locations"
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used **compiler annotation specific memory locations** Found **4,315** of **160,906**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 A real-time microprocessor debugging technique
Charles R. Hill
March 1983 **Proceedings of the symposium on High-level debugging**, Volume 18 , 8 Issue 8 , 4
Full text available: pdf(380.29 KB) Additional Information: full citation, abstract, references

This note describes RED, a remotely executed debugger capable of generating a real-time source level trace history of a high level language program executing on a microprocessor. The trace history consists of a display of the source statements of each basic block executed, annotated by the time at which execution of that block began. Basic blocks are traced rather than statements to reduce sampling bandwidth requirements while still retaining the ability to record the essential logical flow of p ...

2 Ada development system technical and performance requirements (with rationale)
Donald G. Krantz
December 1990 **Proceedings of the conference on TRI-ADA '90**
Full text available: pdf(1.85 MB) Additional Information: full citation, abstract, references

This paper discusses requirements for Ada1 compilers and associated tools used for real-time embedded weapons systems (EWS) development. The requirements have been developed over a period of several years by embedded systems developers at Honeywell Inc. and Alliant Techsystems Inc. Requirements for the run time system, compiler-generated code, and host tools such as linkers are presented. A short rationale statement is provided with each specific requirement.

3 Phase coupling for horizontal microcode generation
Vicki H. Allan, Robert A. Mueller
December 1987 **Proceedings of the 20th annual workshop on Microprogramming**
Full text available: pdf(1.16 MB) Additional Information: full citation, abstract, references, citings

Microcode for mass produced architectures is still predominantly generated by hand. Yet, as speed dictates the vertical migration of commonly executed functions to microcode, the demand for automated code generation increases. Though considerably more complex than phase-decoupled methods, phase-coupled methods for the generation of horizontal microcode have the potential to produce more highly optimized microcode. Results of the retargetable phase-coupled microcode compiler, Hori ...

4 ParADE: An OpenMP Programming Environment for SMP Cluster Systems

Yang-Suk Kee, Jin-Soo Kim, Soonhoi Ha
November 2003 **Proceedings of the 2003 ACM/IEEE conference on Supercomputing**

Full text available: pdf(244.73 KB) Additional Information: full citation, abstract

Demand for programming environments to exploit clusters of symmetric multiprocessors (SMPs) is increasing. In this paper, we present a new programming environment, called ParADE, to enable easy, portable, and high-performance programming on SMP clusters. It is an OpenMP programming environment on top of a multi-threaded software distributed shared memory (SDSM) system with a variant of home-based lazy release consistency protocol. To boost performance, the runtime system provides explicit messag ...

Keywords: programming environment, SMP cluster, software distributed shared memory, hybrid programming, OpenMP, MPI

5 Using annotations to reduce dynamic optimization time

Chandra Krintz, Brad Calder
May 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2001 conference on Programming language design and implementation**, Volume 36 Issue 5

Full text available: pdf(1.78 MB) Additional Information: full citation, abstract, references, citings, index terms

Dynamic compilation and optimization are widely used in heterogenous computing environments, in which an intermediate form of the code is compiled to native code during execution. An important trade off exists between the amount of time spent dynamically optimizing the program and the running time of the program. The time to perform dynamic optimizations can cause significant delays during execution and also prohibit performance gains that result from more complex optimization.

6 Java annotation-aware just-in-time (AJIT) complilation system

Ana Azevedo, Alex Nicolau, Joe Hummel
June 1999 **Proceedings of the ACM 1999 conference on Java Grande**

Full text available: pdf(1.26 MB) Additional Information: full citation, references, citings, index terms

7 An annotation language for optimizing software libraries

Samuel Z. Guyer, Calvin Lin
December 1999 **ACM SIGPLAN Notices , Proceedings of the 2nd conference on Domain-specific languages**, Volume 35 Issue 1

Full text available: pdf(1.34 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper introduces an annotation language and a compiler that together can customize a library implementation for specific application needs. Our approach is distinguished by its ability to exploit high level, domain-specific information in the customization process. In particular, the annotations provide semantic information that enables our compiler to analyze and optimize library operations as if they were primitives of a domain-specific language. Thus, our approach yields many of the ...

8 A certifying compiler for Java

Christopher Colby, Peter Lee, George C. Necula, Fred Blau, Mark Plesko, Kenneth Cline
May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation**, Volume 35 Issue 5

Full text available: pdf(792.48 KB) Additional Information: full citation, abstract, references, citings, index

terms

This paper presents the initial results of a project to determine if the techniques of proof-carrying code and certifying compilers can be applied to programming languages of realistic size and complexity. The experiment shows that: (1) it is possible to implement a certifying native-code compiler for a large subset of the Java programming language; (2) the compiler is freely able to apply many standard local and global optimizations; and (3) the PCC bina ...



9 USC: a universal stub compiler

Sean O'Malley, Todd Proebsting, Allen Brady Montz

October 1994 **ACM SIGCOMM Computer Communication Review , Proceedings of the conference on Communications architectures, protocols and applications**, Volume 24 Issue 4

Full text available: pdf(1.23 MB) Additional Information: full citation, abstract, references, citings, index terms

USC is a new stub compiler that generates stubs that perform many data conversion operations. USC is flexible and can be used in situations where previously only manual code generation was possible. USC generated code is up to 20 times faster than code generated by traditional argument marshaling schemes such as ASN.1 and Sun XDR. This paper presents the design of USC and a comprehensive set of experiments that compares USC performance with the best manually generated code and traditional s ...



10 The design and implementation of a certifying compiler

George C. Necula, Peter Lee

May 1998 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1998 conference on Programming language design and implementation**, Volume 33 Issue 5

Full text available: pdf(1.66 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper presents the design and implementation of a compiler that translates programs written in a type-safe subset of the C programming language into highly optimized DEC Alpha assembly language programs, and a *certifier* that automatically checks the type safety and memory safety of any assembly language program produced by the compiler. The result of the certifier is either a formal proof of type safety or a counterexample pointing to a potential violation of the type system by the t ...



11 Object-oriented programming languages and systems (OOP): [a]C#: C# with a customizable code annotation mechanism

Walter Cazzola, Antonio Cisternino, Diego Colombo

March 2005 **Proceedings of the 2005 ACM symposium on Applied computing**

Full text available: pdf(133.51 KB) Additional Information: full citation, abstract, references

Reflective programming is becoming popular due to the increasing set of dynamic services provided by execution environments like JVM and CLR. With custom attributes Microsoft introduced an extensible model of reflection for CLR: they can be used as additional decorations on element declarations. The same notion has been introduced in Java 1.5. The extensible model proposed in both platforms limits annotations to class members. In this paper we describe [a]C#,[1] an extension of the C# p ...

Keywords: .NET, C#, code annotation, reflection



12 Inferring annotated types for inter-procedural register allocation with constructor flattening

Torben Amtoft, Robert Muller

January 2003 **ACM SIGPLAN Notices , Proceedings of the 2003 ACM SIGPLAN international workshop on Types in languages design and implementation**, Volume 38 Issue 3

Full text available: pdf(268.82 KB) Additional Information: full citation, abstract, references, index terms

We introduce an annotated type system for a compiler intermediate language. The type system is designed to support inter-procedural register allocation and the representation of tuples and variants directly in the register file. We present an algorithm that generates constraints for assigning annotations, and prove its soundness with respect to the type system.

Keywords: certifying compilers, defunctionalization, effects, register allocation, type systems

13 Annotation-directed run-time specialization in C

Brian Grant, Markus Mock, Matthai Philipose, Craig Chambers, Susan J. Eggers

December 1997 **ACM SIGPLAN Notices , Proceedings of the 1997 ACM SIGPLAN symposium on Partial evaluation and semantics-based program manipulation**, Volume 32 Issue 12

Full text available: pdf(1.99 MB) Additional Information: full citation, abstract, references, citings, index terms

We present the design of a dynamic compilation system for C. Directed by a few declarative user annotations specifying where and on what dynamic compilation is to take place, a binding time analysis computes the set of run-time constants at each program point in each annotated procedure's control flow graph; the analysis supports program-point-specific polyvariant division and specialization. The analysis results guide the construction of a specialized run-time specializer for each dynamically c ...

14 Automatic determination of recommended test combinations for Ada compilers

James S. Marr, Patricia K. Lawlis

June 1991 **Proceedings of the eighth annual Washington Ada symposium & summer SIGAda meeting on Ada: software: foundation for competitveness**

Full text available: pdf(756.17 KB) Additional Information: full citation, abstract, references, index terms

Ada compilers are validated using the Ada Compiler Validation Capability (ACVC) test suite, containing over 4000 individual test programs. Each test program focuses, to the extent possible, on a single language feature. Despite the advantages of this "atomic testing" methodology, it is often the unexpected interactions between language features that result in compilation problems. This research investigated techniques to automatically identify recommended combinations of Ada lan ...

15 Points-to analysis for Java using annotated constraints

Atanas Rountev, Ana Milanova, Barbara G. Ryder

October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: pdf(263.51 KB) Additional Information: full citation, abstract, references, citings, index terms

The goal of point-to analysis for Java is to determine the set of objects pointed by a reference variable or a reference object field. This information has a wide variety of client applications in optimizing compilers and software engineering tools. In this paper we present a point-to analysis for Java based on Andersen's point-to analysis for C [5]. We implement the analysis by using a constraint-based approach which employs *annotated inclusion constraints*. Constraint annotations allow u ...

16 A note on the detection of an Ada compiler bug while debugging an Anna program
S. Sankar
June 1989 **ACM SIGPLAN Notices**, Volume 24 Issue 6
Full text available: pdf(570.77 KB) Additional Information: full citation, citings, index terms

17 An optimizing compiler for lexically scoped LISP
Rodney A. Brooks, Richard P. Gabriel, Guy L. Steele
June 1982 **ACM SIGPLAN Notices , Proceedings of the 1982 SIGPLAN symposium on Compiler construction**, Volume 17 Issue 6
Full text available: pdf(1.37 MB) Additional Information: full citation, abstract, references, citings, index terms

We are developing an optimizing compiler for a dialect of the LISP language. The current target architecture is the S-I, a multiprocessing supercomputer designed at Lawrence Livermore National Laboratory. While LISP is usually thought of as a language primarily for symbolic processing and list manipulation, this compiler is also intended to compete with the S-1 PASCAL and FORTRAN compilers for quality of compiled numerical code. The S-1 is designed for extremely high-speed signal processing ...

18 Using visualization tools to teach compiler design
Steven R. Vegdahl
October 2000 **Journal of Computing Sciences in Colleges**, Volume 16 Issue 2
Full text available: pdf(328.42 KB) Additional Information: full citation, abstract, references, citings, index terms

A project-based compiler course presents several challenges to the student-implementor. In addition to the "book learning about" various compiler topics, a student must assimilate a large amount of information about the compiler's implementation. Furthermore he or she must be able understand each source-program construct at a number of different representation levels. Finally, the student must apply that knowledge during implementation and debugging of a compiler.

This pap ...

Keywords: Java, abstract syntax tree, compiler, visualization

19 Regions, compiler support: Region analysis and transformation for Java programs
Sigmund Cherem, Radu Rugina
October 2004 **Proceedings of the 4th international symposium on Memory management**
Full text available: pdf(261.81 KB) Additional Information: full citation, abstract, references, index terms

This paper presents a region analysis and transformation framework for Java programs. Given an input Java program, the compiler automatically translates it into an equivalent output program with region-based memory management. The generated program contains statements for creating regions, allocating objects in regions, removing regions, and passing regions as parameters. As a particular case, the analysis can enable the allocation of objects on the stack.

Our algorithm uses a flow-ins ...

Keywords: pointer analysis, program transformations, region-based memory management

20 Annotated syntax graphs



Samuel Hsieh
October 2002 **Journal of Computing Sciences in Colleges**, Volume 18 Issue 1

Full text available: pdf(42.02 KB) Additional Information: full citation, abstract, references, index terms

Annotated syntax graphs are presented as a pedagogical tool in teaching compiler construction and as a design tool for developing semantic actions in writing a compiler. The tool is compared with attribute grammars. Using annotated syntax graphs generally leads to rather lucid semantic actions. Certain difficulties that exist when using attribute grammars can be avoided when annotated syntax graphs are used instead.

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

Google Web Images Groups News Froogle Local more »
visual basic annotations memory locations Search Advanced Search Preferences

Web Results **1** - **10** of about **214,000** for **visual basic annotations memory locations**. (**0.20** seconds)

Glossary
... in a variety of programming languages, including C, C++, **Visual Basic**, and Java. ... n. The total range of **memory locations** addressable by a computer. ...
support.microsoft.com/?scid=http:/ /support.microsoft.com%2Fsupport%2Fglossary%2Fa.asp - Similar pages

The Code Project - Free Source Code and Tutorials
A simple class written in **Visual Basic** .NET which enables disabled controls on ... NET code programmatically, in **memory**, then use the resulting assembly to ...
www.codeproject.com/vb/net/ - 101k - Sep 12, 2005 - Cached - Similar pages

C++ Tutorials
Pointers...access **memory locations**, Pointers...access **memory locations** ... Beginning **Visual Basic** · Learn VB.net · Learn VB 6 · VB and databases ...
www.programmershelp.co.uk/c++tutorials.php - 60k - Sep 13, 2005 - Cached - Similar pages

CGI Annotation Server Example
... the **annotations** are stored in a structure which mirrors the **location** of the images. ... Contains the **Visual Basic** source code used to create AnnCGI.dll. ...
www.pegasusimaging.com/PrizmViewerDocs/web-server.htm - 27k - Sep 12, 2005 - Cached - Similar pages

[PPT] Slide 1
File Format: Microsoft Powerpoint 97 - View as HTML
Pointer contains address of **memory location**. Pointers are typed ... Derived from it via **Visual Basic**. Derive again from Managed C++. Grandfather in C# ...
docs.msdnaa.net/ark_new/Webfiles/ Courses/DNRK/Common_Type_System.ppt - Similar pages

Introduction to COBOL
Two recent challengers though, Java and **Visual Basic**, are proving to be ... the first number and store it in the named **memory location** (a variable) - Num1 ...
www.csis.ul.ie/COBOL/Course/COBOLIntro.htm - 72k - Cached - Similar pages

AddressOf.com
NET Clients · Differences Between **Visual Basic** .NET and **Visual** C# NET Coding Guidelines to include a few **annotations** that help to answer a few ...
addressof.com/blog/ - 60k - Sep 12, 2005 - Cached - Similar pages

[PDF] Advantage Enterprise
File Format: PDF/Adobe Acrobat - View as HTML
Generate **visual** and/or textual navigation. directions from one **location** to ... The **memory** values listed in Table 2 permit **basic** system evaluation and ...
cdn.mapquest.com/corpb2b/ bsolutions-advantageenterprise-pdf-whitepaper.pdf - Similar pages

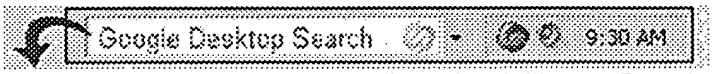
Introduction to Java
In Java, you use primitive types to handle **basic** values and Java objects to ... Note that in Java, the **memory** size of the primitive types is well-defined. ...

Sponsored Links
Annotation OCX/ActiveX
MetaDraw
Download a Free Demo Now
www.Bennet-Tec.com

e-docs.bea.com/workshop/docs81/doc/
en/workshop/guide/getstarted/ovwWhatIfIDontKnowJava.html - 24k - Sep 13, 2005 -
Cached - Similar pages

[poc] Product Information
File Format: Microsoft Word 2000 - View as HTML
MapPoint and Pocket Streets now check the GPS device for **location** ... a more robust object model based on the familiar **Visual Basic** for Applications. ...
web01.microsoft.se/portfolio/product%20information%20sheets/ MapPoint%202004,%20updated%202005-03-01.doc - Similar pages

Gooooooooooogle ▶
Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**



Free! Instantly find your email, files, media and web history. Download now.

visual basic annotations memory loc Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google

©2005 Google

Google Web Images Groups News Froogle Local more »
visual basic annotations memory locations Search Advanced Search Preferences

Web Results **1** - **10** of about **214,000** for **visual basic annotations memory locations**. (**0.20** seconds)

Glossary
... in a variety of programming languages, including C, C++, **Visual Basic**, and Java. ... n. The total range of **memory locations** addressable by a computer. ...
support.microsoft.com/?scid=http:/ /support.microsoft.com%2Fsupport%2Fglossary%2Fa.asp - Similar pages

The Code Project - Free Source Code and Tutorials
A simple class written in **Visual Basic** .NET which enables disabled controls on ... NET code programmatically, in **memory**, then use the resulting assembly to ...
www.codeproject.com/vb/net/ - 101k - Sep 12, 2005 - Cached - Similar pages

C++ Tutorials
Pointers...access **memory locations**, Pointers...access **memory locations** ... Beginning **Visual Basic** · Learn VB.net · Learn VB 6 · VB and databases ...
www.programmershelp.co.uk/c++tutorials.php - 60k - Sep 13, 2005 - Cached - Similar pages

CGI Annotation Server Example
... the **annotations** are stored in a structure which mirrors the **location** of the images. ... Contains the **Visual Basic** source code used to create AnnCGI.dll. ...
www.pegasusimaging.com/PrizmViewerDocs/web-server.htm - 27k - Sep 12, 2005 - Cached - Similar pages

[PPT] Slide 1
File Format: Microsoft Powerpoint 97 - View as HTML
Pointer contains address of **memory location**. Pointers are typed ... Derived from it via **Visual Basic**. Derive again from Managed C++. Grandfather in C# ...
docs.msdnaa.net/ark_new/Webfiles/ Courses/DNRK/Common_Type_System.ppt - Similar pages

Introduction to COBOL
Two recent challengers though, Java and **Visual Basic**, are proving to be ... the first number and store it in the named **memory location** (a variable) - Num1 ...
www.csis.ul.ie/COBOL/Course/COBOLIntro.htm - 72k - Cached - Similar pages

AddressOf.com
NET Clients · Differences Between **Visual Basic** .NET and **Visual** C# NET Coding Guidelines to include a few **annotations** that help to answer a few ...
addressof.com/blog/ - 60k - Sep 12, 2005 - Cached - Similar pages

[PDF] Advantage Enterprise
File Format: PDF/Adobe Acrobat - View as HTML
Generate **visual** and/or textual navigation. directions from one **location** to ... The **memory** values listed in Table 2 permit **basic** system evaluation and ...
cdn.mapquest.com/corpb2b/ bsolutions-advantageenterprise-pdf-whitepaper.pdf - Similar pages

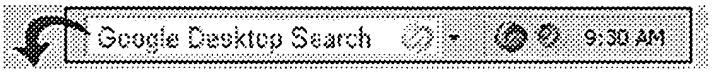
Introduction to Java
In Java, you use primitive types to handle **basic** values and Java objects to ... Note that in Java, the **memory** size of the primitive types is well-defined. ...

Sponsored Links
Annotation OCX/ActiveX
MetaDraw
Download a Free Demo Now
www.Bennet-Tec.com

e-docs.bea.com/workshop/docs81/doc/
en/workshop/guide/getstarted/ovwWhatIfIDontKnowJava.html - 24k - Sep 13, 2005 -
Cached - Similar pages

[DOC] Product Information
File Format: Microsoft Word 2000 - View as HTML
MapPoint and Pocket Streets now check the GPS device for **location** ... a more robust object model based on the familiar **Visual Basic** for Applications. ...
web01.microsoft.se/portfolio/product%20information%20sheets/ MapPoint%202004,%20updated%202005-03-01.doc - Similar pages

Gooooooooooogle ▶
Result Page: 1 2 3 4 5 6 7 8 9 10 **Next**



Free! Instantly find your email, files, media and web history. Download now.

visual basic annotations memory loc Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

Google Home - Advertising Programs - Business Solutions - About Google

©2005 Google